Zaawansowane aplikacje systemów sztucznej inteligencji

Przedstaw 3 różne zastosowania sieci neuronowej typu PNN

Bartosz Ociepka

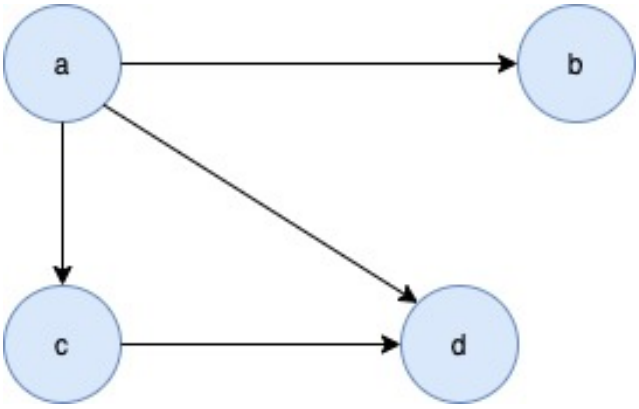
21 stycznia 2021

Opis historyczny (geneza)

Pierwsza probablistyczna sieć neuronowa powstała w 1966 roku i została opisana przez doktora Donalda F. Sprechta w dziele "Generation of Polynomial Discriminant Functions for Pattern Recognition". Ten nowy rodzaj sieci wywodzi się z sieci bayesowskiej i metoda wyznaczania funkcji dyskryminacyjnej Fishera oparta na funkcji jądra (ang. Kernel Fisher Discriminant, KFD). Sieć bayesowska jest pewnego rodzaju modelem służącym do przedstawiania zależności między zdarzeniami. Zwykle sieć bayesowską przedstawia się jako skierowany graf acykliczny, gdzie wierzchołki to zdarzenia, a łuki to związki między nimi. Jeśli dwa zdarzenia nie są połączone żadną ścieżką to te zdarzenia na siebie nie wpływają. W opisie matematycznym będziemy mogli zauważyć podobieństwa sieci bayesowkiej i PNN.

Opis matematyczny zależności wraz ze schematem poglądowym

Opis matematyczny zacznijmy od określenia sieci PNN jako sieci jednokierunkowej (feed-forward). Oznacza to, że informacje w ramach sieci przepływają tylko w jedną stronę (odwrotnie niż w przypadku np. sieci rekurencyjnych). Jako, że sieć jest skierowana to nie możemy zdefiniować w niej cykli.



Rysunek 1: Przykład grafu skierowanego

Najczęstszym użyciem PNN są problemy klasyfikacyjne. Probabilistyczna sieć neuronowa potrafi na podstawie zbioru uczącego (w którym mamy przypisanie do konkretnych klas) przewidzieć wynik dla konkretnych danych. Przykładem takiego problemu klasyfikacyjnego może być np. przewidywanie cukrzycy u indian na podstawie pewnych danych o ich pozostałych zdrowotnych aspektach https://www.kaggle.com/uciml/pima-indians-diabetes-database).

Sieć PNN składa się z 4 warstw: warstwa wejściowa, warstwa wzorców, warstwa sumowania i warstwa wyjściowa.

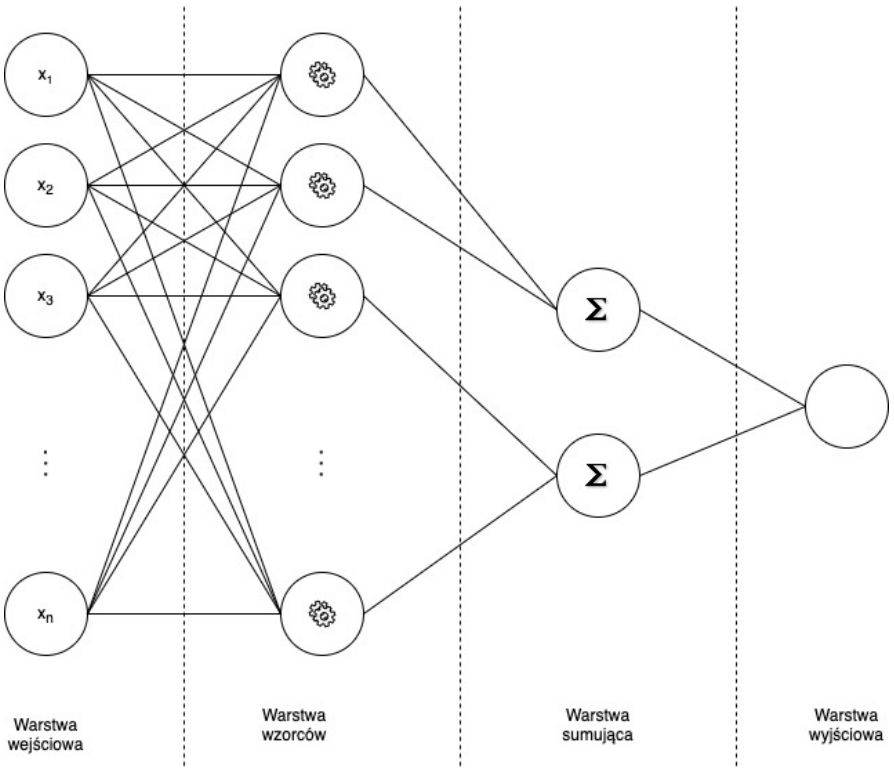
Warstwa wejściowa dokonuje wstępnej obróbki danych. W tej warstwie każdy neuron reprezentuje inną zmienną predykcyjną.
Warstwa wzorców posiada jeden neuron dla każdgo elementu w zbiorze uczącym. Każdy z

neuronów oblicza na podstawie danych wejściowych wynik następującej funkcji decyzyjnej.

$$exp[-(\frac{d(x, x_i)}{p\sigma})^2]$$

gdzie
d jest odległością euklidesową pomiędzy wektorami x oraz x_i reprezentującymi odpowiednio obiekt identyfikowany oraz i-ty przykład uczący
n jest liczbą wektorów uczących danej kategorii
p jest rozmiarem wektora x oraz wszystkich wektorów uczących x_i
σ jest parametrem kształtu

Warstwa sumująca posiada jeden neuron dla każdej kategorii, do której może my przypisać. Neurony w tej warstwie sumują wartości dla każdej kategorii na podstawie wyników funkcji decyzyjnej z warstwy wzorców.
Warstwa wyjściowa bierze wyniki warstwy sumującej i na podstawie tych danych przewiduje wynik.



Rysunek 2: Przykład modelu sieci probabilistycznej

Istnieje możliwość połączenia warstwy wzorców i warstwy sumowania lecz standardowy model sieci PNN zawiera podział na te 2 warstwy.

Uczenie sieci odbywa się na zasadzie odwzorowania całego zbioru uczącego. Przy dodawaniu nowego elementu w zbiorze uczącym to należy przyporządkować go do danej klasy abstrakcji (możemy tutaj np. użyć metody centroidów, przykład użycia w przykładnie nr 1) oraz powiększamy warstwę wzorców o dodatkowy neuron. Odrzucenie elementu ze zbioru uczącego spowoduje usunięcie odpowiadającego neuronu z warstwy wzorców.

Przykład użycia nr 1

Przykładem użycia niech będzie OCR (ang. optical character recognition) czyli optyczne rozpoznawanie znaków. Nasze znaki będą składać się z 2 wartości x i y odpowiednio długość i szerokość. Przykładowy zbiór testowy z literami O, X i I mógłby wyglądać następująco (dane losowe jedynie dla zobrazowania tego jak działa sieć):

1. O- > (0.5, 0.7)
2. o -> (0.2, 0,5)
3. X -> (0.8, 0.8)
4. x -> (0.4, 0.5)
5. l -> (0.6, 0.3)
6. i -> (0.3, 0.2)

Dla naszego wektora wejściowego funkcja aktywacji będzie wyglądała następująco:

$$f(x,y) = e^{\frac{-((x-x_0)^2+(y-y_0)^2)}{2\cdot 0.1^2}}$$

W PNN nie ma uczenia więc od razu możemu przystąpić do przypisania do kategoryzacji. Sprobujmy przypisać lekko zniekształconą literę O o wektorze (0.2, 0.6), do którejś kategorii. W ramach warstwy wzorców jest 6 neuronów (po jednym dla każdego wektora w zbiorze uczącym), a ich wyniki to:

$$f_1 = e^{\frac{-((0.2-0.5)^2+(0.6-0.7)^2)}{2*0.1^2}} = 0.00673795$$

$$f_2 = e^{\frac{-((0.2-0.3)^2+(0.6-0.6)^2)}{2*0.1^2}} = 0.606531$$

$$f_3 = e^{\frac{-((0.2-0.8)^2+(0.6-0.8)^2)}{2*0.1^2}} = 2 \cdot 10^-9$$

$$f_4 = e^{\frac{-((0.2-0.4)^2+(0.6-0.5)^2)}{2*0.1^2}} = 0.0820850$$

$$f_5 = e^{\frac{-((0.2-0.6)^2+(0.6-0.3)^2)}{2*0.1^2}} = 3 \cdot 10^-6$$

$$f_6 = e^{\frac{-((0.2-0.3)^2+(0.6-0.2)^2)}{2*0.1^2}} = 0.000203468$$

Następnie w ramach warstwy sumującej sumujemy wyniki neuronów wchodzących w skład tej samej kategorii. W naszym wypadku wyniki (zaokrąglone do 2 miejsc po przecinku) będą następujące:

$$f_O \approx 0.61$$

$$f_X \approx 0.08$$

$$f_I \approx 0.00$$

Warstwa wyjściowa bierze największy wynik z warstwy sumującej i do odpowiadającej kategorii przypisuje wektor wejściowy - w naszym wypadku będzie to litera 'O'.
Metoda k-means (inaczej metoda centroidów) jest jedną z metod służących do klasteryzacji danych w uczeniu nienadzorowanym. Składa się ona z kilku kroków:

1. Ustalamy liczbę centroidów czyli liczba klas abstrakcji w naszym zbiorze testowym (dla naszego przykładu będą 3 skupienia)
2. Ustalamy wstępne środki centroidów (jako że mamy już przyporządkowanie do klas to naszymi centroidami będą współrzędne będące średnią arytmetyczną współrzędnych należących do danej klasy)
3. Obliczamy odległość nowego obiektu od centroidów
4. Na podstawie wyników z etapu 3. przyporządkowujemy nowy obiekt do klasy abstrakcji
5. Ustalamy nowe środki centroidów jako że zbiór nam się powiększył

Więc dla naszego przypadku z OCR ustalamy 3 klasy abstrakcji - O, X oraz I. Nasze centroidy będą miały współrzędne

- C_O -> (0.35, 0.6)
- C_X -> (0.6, 0.65)
- C_I -> (0.45, 0.25)

Załóżmy, że chcielibyśmy dodać do naszego zbioru punkt X -> (0.4, 0.5). Odległości punktu X od centroidów będą kształtować się następująco:

$$d(X, C_O) = \sqrt{(0.35 - 0.4)^2 + (0.6 - 0.5)^2} \approx 0.111$$

$$d(X, C_X) = \sqrt{(0.6 - 0.4)^2 + (0.65 - 0.5)^2} = 0.25$$

$$d(X, C_I) = \sqrt{(0.45 - 0.4)^2 + (0.25 - 0.5)^2} \approx 0.25$$

Punkt X przyporządkowujemy do klasy abstrakcji określającej literę 'O'. Nowy centroid dla tej klasy abstrakcji będzie znajdował się w punkcie C_O -> (0.36, 0.6).

Przykład użycia nr 2

Kolejnym przykładem może być przewidywanie cukrzycy u indian Pima. Zbiór iczący znajduje się na stronie https://www.kaggle.com/uciml/pima-indians-diabetes-database. Każdy element zbioru uczącego zawiera nastpujące infromacje:

- ilość ciąż
- stężenie glukozy we krwi

- ciśnienie krwi
- grubość skóry w okolicy tricepsa
- insulina
- indeks BMI
- wynik pewnej funkcji obrazującej historię cukrzycy w rodzinie
- wiek

Takie też będą dane wejściowe sieci neuronowej. Wyjściem będzie liczba 1 lub 0 obrazująca czy przewidujemy cukrzycę czy też nie. W ramach zobrazowania przykładu wyciągnijmy kilka elementów zbioru i na tej podstawie spróbujmy przewidzieć wynik.

1. Cukrzyca stwierdzona
   - 1 -> (6, 148, 72, 35, 0, 33.6, 0.627, 50)
   - 2 -> (8, 183, 64, 0, 0, 23.3, 0.672, 32)
   - 3 -> (0 ,137, 40, 35, 168, 43.1, 2.288, 33)
2. Cukrzyca nie została stwierdzona
   - 4 -> (1, 85, 66, 29, 0, 26.6, 0.351, 31)
   - 5 -> (1, 89, 66, 23, 94, 28.1, 0.167, 21)
   - 6 -> (5, 116, 74, 0, 0, 25.6, 0.201, 30)

Spróbujmy przewidzieć wynik dla danych wejściowych P -> (3, 78, 50, 32, 88, 31, 0.248, 26). Przejdźmy do liczenia:

$$f_1 = e^{\frac{-((3-6)^2+(78-148)^2+(50-72)^2+(32-35)^2+(88-0)^2+(31-33.6)^2+(0.248-0.627)^2+(26-50)^2)}{3*0.1^2}} \approx 5.801 * 10^{-198747}$$

$$f_2 = e^{\frac{-((3-8)^2+(78-183)^2+(50-64)^2+(32-0)^2+(88-0)^2+(31-23.3)^2+(0.248-0.672)^2+(26-32)^2)}{3*0.1^2}} \approx 4.055 * 10^{-291115}$$

$$f_3 = e^{\frac{-((3-0)^2+(78-137)^2+(50-40)^2+(32-35)^2+(88-168)^2+(31-43.1)^2+(0.248-1.188)^2+(26-33)^2)}{3*0.1^2}} \approx 3.583 * 10^{-147640}$$

$$f_4 = e^{\frac{-((3-1)^2+(78-85)^2+(50-66)^2+(32-29)^2+(88-0)^2+(31-26.6)^2+(0.248-0.351)^2+(26-31)^2)}{3*0.1^2}} \approx 1.844 * 10^{-117352}$$

$$f_5 = e^{\frac{-((3-1)^2+(78-89)^2+(50-66)^2+(32-23)^2+(88-94)^2+(31-28.1)^2+(0.248-0.167)^2+(26-21)^2)}{3*0.1^2}} \approx 9.064 * 10^{-7694}$$

$$f_6 = e^{\frac{-((3-5)^2+(78-116)^2+(50-74)^2+(32-0)^2+(88-0)^2+(31-25.6)^2+(0.248-0.201)^2+(26-30)^2)}{3*0.1^2}} \approx 1.020 * 10^{-156884}$$

W ramach warstwy sumującej wyniki to:

$$f_{cukrzyca} \approx 3.583 * 10^{-147640}$$

$$f_{brakcukrzycy} \approx 9.064 * 10^{-7694}$$

Przewidujemy więc brak cukrzycy.

Inne przykłady użycia

- Identyfikacja statków [3]
- Klasyfikacja terenu na podstawie zdjęć satelitarnych [4]
- Rozpoznawanie emocji [5]

Literatura źródłowa

1. Specht, D. F. (1967-06-01). "Generation of Polynomial Discriminant Functions for Pattern Recognition". IEEE Transactions on Electronic Computers. EC-16 (3): 308–319. doi:10.1109/PGEC.1967.264667. ISSN 0367-7508.
2. Specht, D. F. (1990). "Probabilistic neural networks". Neural Networks. 3: 109–118. doi:10.1016/0893-6080(90)90049-Q
3. http://www.iaeng.org/publication/IMECS2009/IMECS2009_pp1291-1294.pdf
4. https://www.researchgate.net/publication/272199040_Using_an_unsupervised_approach_of_Probabilistic_Neural_Network_PNN_for_land_use_classification_from_multitemporal_satellite_images
5. https://ieeexplore.ieee.org/document/7844584
6. https://www.logistyka.net.pl/bank-wiedzy/item/download/74774_8d3c3b6c9d642a1435c23f79c2b4eff0